WHAT IS CLAIMED IS:

1. A multiprocessor computer system comprising:

a plurality of processor nodes;

a memory, wherein the memory includes a plurality of lines and a cache coherence directory structure, wherein the plurality of lines includes a first line and wherein the cache coherence directory structure includes a plurality of directory structure entries; and

an interconnect network connecting the plurality of processor nodes to the memory;

wherein each directory structure entry includes processor pointer information indicating the processor nodes that have cached copies of the first line, wherein the processor pointer information includes a plurality n of bit vectors, where n is an integer greater than one;

wherein the n bit vectors define a matrix having a number of locations equal to the product of the number of bits in each of the n bit vectors, wherein the number of locations is at least equal to the number of processor nodes and wherein each of the processor nodes is mapped to a corresponding one of the locations wherein the locations corresponding to the processor nodes are dispersed in the matrix in an at least partially noncontiguous manner.

2. The multiprocessor computer system according to claim 1, wherein each of the processor nodes is assigned a corresponding processor number, wherein each processor number is expressed as n sets of bits, wherein the n sets of bits do not overlap.

3. The multiprocessor computer system according to claim 2, wherein each processor node includes a plurality of processors.

4. The multiprocessor computer system according to claim 1, wherein each processor node is assigned a processor number, wherein the processor number is expressed as a function of a first and a second set of bits; and

wherein the n bit vectors include a first and a second bit vector, wherein the first set of bits are mapped into the first bit vector and wherein the second bit vector is a function of at least a subset of the first set of bits of the processor number and at least a subset of the second set of bits.

5. The multiprocessor computer system according to claim 4, wherein the n bit vectors include a third bit vector which is a function of at least a subset of the second set of bits of the processor number.

6. The multiprocessor computer system according to claim 5, wherein at least one bit of the second bit vector is a function of a bit from the third bit vector and a bit from the first bit vector.

7. The multiprocessor computer system according to claim 6, wherein the bit of the second bit vector is determined by an exclusive or (XOR) operation of the bit from the third bit vector and the bit from the first bit vector.

8. The multiprocessor computer system according to claim 6, further comprising a bit mode field configured to set a functional relationship between the bits of the processor node number and the bits of the n bit vectors.

9. A multiprocessor computer system comprising:

a plurality of processor nodes;

a memory, wherein the memory includes a plurality of lines and a cache coherence directory structure, wherein the plurality of lines includes a first line and wherein the cache coherence directory structure includes a plurality of directory structure entries; and

an interconnect network connecting the plurality of processor nodes to the memory;

wherein each of the processor nodes is assigned a corresponding processor number, wherein the processor number is defined by a four bit N0, N1, N2, N3 subfield of a seven bit field and a three bit N4, N5, N6 subfield of the seven bit field, where the four and three bit subfields do not overlap and where the bits are ordered such that N0 is a least significant bit and N6 is a most significant bit of the seven bit field;

wherein each directory structure entry includes processor pointer information indicating the processor nodes that have cached copies of the first line, wherein the processor pointer information includes a first bit vector which includes sixteen bits, a second bit vector which includes eight bits and a third bit vector which includes two bits;

wherein the first bit vector is defined by bits N0, N1, N2 and N3;

wherein when the number of processor nodes is between one and thirty-two,the second bit vector is

defined by N3, N2 and the result of an XOR operation between N4 and N1;and the third bit vector is defined by N4;

wherein when the number of processor nodes is between thirty-three and sixty-four,the second bit vector is defined by N3, the result of an XOR operation between N5 and N2, and N4, and the third bit vector is defined by N5;

wherein when the number of processor nodes is between sixty-five and 128,the second bit vector is defined by the result of an XOR operation between N6 and N3,N5, and N4, and the third bit vector is defined by N6;

wherein when the number of processor nodes is between 129 and 256,the processor number is defined by an eight bit field which includes the seven bit field and an additional bit N7, the second bit vector is defined by N6, N5 and N4,and the third bit vector is defined by N7.

10. The multiprocessor computer system according to claim 9 further comprising a sharing mode field configured to define a range of the number of processor nodes, wherein a first range has between one and thirty-two processor nodes, a second range has between thirty-three and sixty-four processor nodes, a third range has between sixty-five and 128 processor nodes and a fourth range has between 129 and 256 processor nodes.

11. A method of maintaining cache coherency across a computer system having a plurality of processor nodes, including a first and a second processor node, wherein each of the plurality of processor nodes includes a cache, the method comprising the steps of:

assigning a processor number to each of the plurality of processor nodes;

defining a plurality of bit vectors, wherein each processor node is represented by a bit in each of the plurality of bit vectors;

defining a matrix from the plurality of bit vectors wherein the matrix has a plurality of potential node locations and wherein the number of potential node locations is greater than the number of processor nodes;

mapping each processor number to a corresponding potential node location defining an actual node location wherein the actual node locations corresponding to the processor numbers are dispersed in the matrix in an at least partially noncontiguous manner;

determining a first memory location in memory;

reading a line from the first memory location into the cache of said first processor node, wherein the step of reading includes the step of setting a bit in each of the plurality of bit vectors as a function of the processor number assigned to the first processor node defining a first actual node location of the actual node locations;

reading a line from the first memory location into the cache of said second processor node, wherein the step of reading includes the step of setting a bit in each of the plurality of bit vectors as function of the processor number assigned to the second processor node defining a

second actual node location of the actual node locations; and

writing to said first memory location, wherein the step of writing includes the steps of:

(a) determining, as a function of bits set in the bit vectors, processor nodes holding a copy of the line; and

(b) sending a message to the processor nodes determined in step (a) indicating that the line read from the first memory location is no longer valid.

12. The method of maintaining cache coherency across a computer system according to claim 11 wherein the step of mapping further comprises:

mapping a first set of bits from the processor number into a first bit vector of the plurality of bit vectors; and

mapping at least a subset of the first set of bits from the processor number and at least a subset from a second set of bits from the processor number into a second bit vector of the plurality of bit vectors.

13. The method of maintaining cache coherency across a computer system according to claim 12 wherein the step of mapping further comprises determining at least one bit of the second bit vector as a function of a bit from the third bit vector and a bit from the first bit vector.

14. The method of maintaining cache coherency across a computer system according to claim 13 wherein the step of mapping further comprises mapping a first set of bits from the processor number into a first bit vector of the plurality of bit vectors.

15. The method of maintaining cache coherency across a computer system according to claim 14 wherein the step of determining further comprises determining the bit of the second bit vector by an exclosuve or (XOR) operation of the bit from the third bit vector and the bit from the first bit vector.

16. The method of maintaining cache coherency across a computer system according to claim 15 further comprising setting a functional relationship between the bits of the processor number and the bits of the plurality of the bit vectors based on the number of processor nodes in the computer system.

17. The method of maintaining cache coherency across a computer system according to claim 16 wherein the first bit vector is an N-to-2^N mapping of the N least significant bits of the processor number.

18. The method of maintaining cache coherency across a computer system according to claim 17 wherein the second bit vector is an M-to-2^M mapping of bits (M+N-1) through N of the processor number.

19. The method of maintaining cache coherency across a computer system according to claim 18 wherein the third bit vector is a P-to-2^P mapping of bits (P+M+N-1) through M+N of the processor number.

20. The method of maintaining cache coherency across a computer system according to claim 19 wherein M=4, N=3 and P=2;

wherein the processor number is defined by an eight bit number N0, N1, N2, N3, N4, N5, N6, and N7;

wherein the first bit vector is defined by bits N0, N1, N2 and N3;

wherein when the number of processor nodes is between one and thirty-two, the second bit vector is defined by N3, N2 and the result of an XOR operation between N4 and N1;and the third bit vector is defined by N4;

wherein when the number of processor nodes is between thirty-three and sixty-four, the second bit vector is defined by N3, the result of an XOR operation between N5 and N2, and N4, and the third bit vector is defined by N5;

wherein when the number of processor nodes is between sixty-five and 128,the second bit vector is defined by the result of an XOR operation between N6 and N3,N5, and N4, and the third bit vector is defined by N6;

wherein when the number of processor nodes is between 129 and 256,the processor number is defined by an eight bit field which includes the seven bit field and an additional bit N7, the second bit vector is defined by N6, N5 and N4,and the third bit vector is defined by N7.